IN THE CLAIMS

1 (Currently Amended). The [[A]] method of claim 29 further comprising:

in a processor based system where a plurality of logical processors of a single physical processor share processor execution resources of the single physical processor, in response to a first logical processor in the plurality of processors being scheduled to enter an idle state due to lack of scheduling tasks, making a processor execution resource previously reserved for the first logical processor available to any of the plurality of logical processors.

2 (Original). The method of claim 1 further comprising reserving the processor execution resource for the first processor in response to the first processor being scheduled to execute a task.

3 (Original). The method of claim 2 wherein each of the plurality ofprocessors is a logical processor of the processor based system.

4 (Original). The method of claim 3 wherein the first processor being scheduled to enter an idle state further comprises the first processor executing a processor instruction requesting the first processor to enter an idle state.

5 (Previously Presented). The method of claim 4 wherein making the processor execution resource previously reserved for the first processor available to any of the plurality of processors further comprises releasing the processor execution resource into a common pool of processor execution resources.

6 (Previously Presented). The method of claim 2 wherein the first processor being scheduled to execute a task further comprises the first processor receiving a wake up signal.

7 (Previously Presented). The method of claim 6 wherein the processor execution resource previously reserved for the first processor further is statically allocated to the first processor; and wherein releasing the processor execution resource into a common pool of

processor execution resources further comprises de-allocating the processor execution resource from the first processor.

8 (Previously Presented). The method of claim 6 wherein the processor execution resource previously reserved for the first processor is locked by the first processor; and

wherein releasing the processor execution resource into a common pool of processor execution resources further comprises unlocking the processor execution resource.

9 (Previously Presented). The method of claim 5 wherein the common pool of processor execution resources comprises a translation lookaside buffer and the processor execution resource is a translation cache entry from the translation lookaside buffer.

Claims 10-19 (Canceled).

20 (Currently Amended). The [[A]] machine accessible medium of claim 30 further ~~having stored thereon data which when accessed by a machine causes the machine to perform a method, the method~~ comprising:

in a processor based system where a plurality of logical processors implemented in a single physical processor share processor execution resources of the physical processor, in response to a first logical processor in the plurality of logical processors being scheduled to enter an idle state due to lack of scheduled tasks, making a processor execution resource previously reserved for the first logical processor available to a second logical processor in the plurality of processors.

21 (Original). The machine accessible medium of claim 20 further comprising reserving the processor execution resource for the first processor in response to the first processor being scheduled to execute a task.

22 (Original). The machine accessible medium of claim 21 wherein each of the plurality of processors is a logical processor of the processor based system.

23 (Original). The machine accessible medium of claim 22 wherein the first processor being scheduled to enter an idle state further comprises the first processor executing a processor instruction requesting the first processor to enter an idle state.

24 (Original). The machine accessible medium of claim 23 wherein making the processor execution resource previously reserved for the first processor available to a second processor further comprises releasing the processor execution resource into a common pool of processor execution resources accessible from the second processor.

25 (Original). The machine accessible medium of claim 24 wherein the first processor being scheduled to execute a task further comprises the first processor receiving a wake up signal.

26 (Original). The machine accessible medium of claim 25 wherein the processor execution resource previously reserved for the first processor further comprises the processor execution resource previously statically allocated to the first processor; and wherein releasing the processor execution resource into a common pool of processor execution resources further comprises de-allocating the processor execution resource.

27 (Original). The machine accessible medium of claim 25 wherein the processor execution resource previously reserved for the first processor further comprises the processor execution resource previously locked by the first processor; and wherein releasing the processor execution resource into a common pool of processor execution resources further comprises the first processor unlocking the processor execution resource.

28 (Original). The machine accessible medium of claim 25 wherein the common pool of processor execution resources comprises a translation lookaside buffer and the processor execution resource is a translation cache entry from the translation lookaside buffer.

29 (New). A method comprising:

in an operating system thread scheduler, detecting that a thread executed by a processor has terminated;

switching the processor out from the ongoing thread;

determining if there are other tasks scheduled for the processor; and

if not, commanding the processor to release any execution resources exclusively held by the processor and to suspend.

30 (New). A machine accessible medium having stored thereon data which, when accessed by a machine, causes the machine to perform a method comprising:

in an operating system thread scheduler, detecting that a thread executed by a processor has terminated;

switching the processor out from the ongoing thread;

determining if there are other tasks scheduled for the processor; and

if not, commanding the processor to release any execution resources exclusively held by the processor and to suspend.